ONDERZOEKSRAPPORT NR. 8901

THE GENERATION OF STRONGLY-RANDOM ACTIVITY NETWORKS

BY

W. HERROELEN, E. DEMEULEMEESTER, B. DODIN

D/1989/2376/1

THE GENERATION OF STRONGLY-RANDOM ACTIVITY NETWORKS

Willy HERROELEN*
Erik DEMEULEMEESTER*
Bajis DODIN**

* Department of Applied Economic Sciences, Katholieke Universiteit Leuven, Dekenstraat 2, B-3000 Leuven, Belgium

** Graduate School of Management, University of California, Riverside, CA 92521, U.S.A.

THE GENERATION OF STRONGLY-RANDOM ACTIVITY NETWORKS

Willy HERROELEN
Erik DEMEULEMEESTER
Bajis DODIN

ABSTRACT

The generation of random test problems is a major concern in the computational experiments which have to be performed during the validation process of optimal and suboptimal solution procedures for the many combinatorial problems in the area of activity networks. The test problems should be random both in terms of network functions (such as activity duration, resource requirements, etc.) and network structure. In this paper we present two procedures for generating a random topological structure of an activity-on-the-arc network with a given number of nodes and arcs. These algorithms are incorporated in a software package for the generation of a set of random activity networks characterized by a suitable range of the number of nodes and arcs and corresponding topological structure.

1. INTRODUCTION

Numerous computational experiments have been conducted in order to validate the optimal and suboptimal procedures which have been developed for the solution of many combinatorial problems in the area of activity networks. A reliable validation procedure commonly implies the solution of a large number of representative problems and relies on such estimates as required computer time and memory to measure algorithm efficiency. The set of representative test problems is either taken from existing problem sets described in the literature (for example the standard set of 110 test problems for resource-constrained project scheduling assembled by Patterson (1984)) or may be obtained by the individual authors through the use of their own (computerized) procedure to generate both the topological structure of the activity networks (that is, the structure of the underlying directed, acyclic graph) and a set of random values representing suitable outcomes for the functions associated with the network (such as activity durations, resource requirements and availabilities, direct activity costs, etc.). Examples of the latter approach can be found in Alvarez-Valdés and Tamarit (1988), Christofides et al. (1987), Dodin (1985), Kurtulus and Davis (1982) and Talbot (1982).

In order to derive reliable results with respect to the impact of network structure on the performance and the computational effort required by optimal and suboptimal solution procedures, it is mandatory that the activity networks included in the computational experiment are indeed characterized by a random topological structure (see also Elmaghraby and Herroelen 1980). The random generation of activity network functions for networks with a given topological structure; i.e., the generation of weakly-random networks, does not pose major problems.

The generation of **strongly-random activity networks**, in which both the functions and the structure are randomly determined, however, proves to be an onerous task. A clear definition of the notion of a random network seems to be missing, resulting in the fact that the construction rules used by many network generators are centered around rather loose statistical objectives.

The purpose of this paper is to present a computerized algorithm for generating a set of strongly-random activity networks. An explicit problem statement is given in the next section. Section 3 will focus on two algorithms for generating a single random activity-on-the-arc network with a given number of nodes and arcs. The overall procedure for generating a set of strongly-random activity networks will then be discussed in Section 4. Since the procedure basically generates a set of strongly-random, acyclic, directed graphs, it's use is not restricted to the narrow field of activity networks.

2. DETAILED PROBLEM STATEMENT

The topological structure of an **activity network** (further abbreviated as AN) consists of an acyclic directed graph $G = (\mathbf{N},\mathbf{A})$, where the symbol **N** represents the set of nodes as well as their count and the symbol **A** represents the set of arcs as well as their count. In the sequel we assume that the AN is in the **activity-on-the-arc** representation; i.e., the nodes represent the network events and the arcs denote the network activities. The nodes are numbered such that an arc always leads from a small number to a larger one, and there is only one start and one end node to the AN. An immediate consequence of such a numbering scheme is that the adjacency matrix is always upper triangular with zero diagonal. A typical AN and corresponding adjacency matrix for $N = 4$ and $A = 5$ are given in Figure 1.

****
Figure 1
****

It should be clear that for a given **N** and **A**, several feasible **G(N,A)** may be generated. Figure 2 lists the other three alternative topological structures for an AN with N = 4 and A = 5.

****
Figure 2
****

Consequently, the generation of a strongly-random **G(N,A)** for fixed **N** and **A** implies that the resultant topological network structures should have equal probabilities of occurrence. In the next section two procedures are given that satisfy this requirement.

3. GENERATING A RANDOM ACTIVITY NETWORK

Generating a network **G(N,A)** such that the various topological network structures which are possible for the given **N** and **A** have equal probabilities of occurrence can be achieved using the two procedures described in this section. The first procedure, denoted as the **Deletion Method**, starts from the completely connected AN; i.e., the adjacency matrix filled with ones in the upper diagonal part, and deletes the necessary number of arcs until the desired number of arcs are left. The second procedure, called the **Addition Method**, starts with the unordered AN and generates the required number of arcs.

3.1 The Deletion Method

Consider the adjacency matrix $[a_{ij}]$ corresponding to a completely connected activity network **G(N,A)**. This adjacency matrix contains N(N-1)/2 ones. Let the outdegree of node i be defined as the total number of arcs leaving the node, and define the indegree of node j as the total number of arcs entering the node. The outdegree of node i can then be written as

$$n_i = \sum_j a_{ij} = N - i , \qquad [1]$$

and the indegree of node j as

$$m_j = \sum_i a_{ij} = j - 1 \qquad [2]$$

The Deletion Method now reduces to the random deletion of $D = N(N-1)/2 - A$ ones in the adjacency matrix, such that

$$n_i >= 1, \quad i = 1,2,\ldots, N-1$$
$$n_i = 0, \quad i = N \qquad [3]$$

and

$$m_j = 0, \quad j = 1$$
$$m_j >= 1, \quad j = 2,3,\ldots, N \qquad [4]$$

For any AN, the above conditions simply state that at least one arc must leave every node except the last, and at least one arc should enter every node except the first.

The Deletion Method should generate activity networks with equal ex ante probabilities for the different feasible topological network structures; i.e., all existing ones in the adjacency matrix for the completely connected network should receive equal deletion probabilities given the consistency constraints given in Eqs. [3] and [4]. This can be achieved by numbering all the ones in the adjacency matrix for the completely connected AN from left to right and consecutively in the rows, as illustrated in Figure 3.

****

Figure 3

****

The corresponding numbers (labels) are then assigned to equal intervals in the range of a uniformly distributed variable. Drawing a random number will now yield an interval which in turn identifies the label of a corresponding arc.

In order to select a particular arc (i^*,j^*) to be deleted, we proceed as follows. The interval corresponding to a node i^* has a length equal to $(N-i^*)$ times the interval length of a label. For example in Figure 3, the interval corresponding to node $i^*=2$ has a length of $(4-2)(1/6) = 1/3$. It can also be seen from Figure 3 that $i^*=2$ is preceded by three intervals, each of length 1/6. In general, node i^* is preceded by at least

$$\sum_{0<i<i^*} (N-i) = (i^*-1)N - i^*(i^*-1)/2 \qquad [5]$$

labeled intervals.

In order to generate an i^*, let $Y \sim U(0,1)$, and let

$$X = Y\,(N(N-1)/2) \qquad [6]$$

where $N(N-1)/2$ denotes the total number of labels (total number of arcs in the completely connected AN). Now the interval relation between X and i^* implies that (see Eq. [5])

$$X >= (i^*-1)N - i^*(i^*-1)/2$$

or with $0 <= \alpha < N-i^*$

$$i^{*2}/2 - (N + 0.5)i^* + (N + X - \alpha) = 0$$

which yields

$$i^* = (N + 0.5) \pm \sqrt{(N + 0.5)^2 - 2(N + X - \alpha)} \qquad [7]$$

Since i^* <= N-1 we must select the negative root. Moreover, since $\alpha >= 0$, Eq. [7] reduces to

$$i^* <= (N + 0.5) - \sqrt{(N + 0.5)^2 - 2N - 2X} \quad ,$$

or

$$i^* <= (N + 0.5) - \sqrt{(N - 0.5)^2 - 2X}$$

Substituting from Eq. [6] yields

$$i^* <= (N + 0.5) - \sqrt{N(N-1)(1-Y) + 0.25}$$

Since Y ~ U(0,1) implies that (1-Y) ~ U(0,1), we have

$$i^* <= N + 0.5 - \sqrt{N(N-1)Y + 0.25} \qquad [8]$$

Since the last label interval corresponding to node i^* is followed by a number of label intervals at most equal to $i^*N - i^*(i^* + 1)/2$, a symmetrical argument leads to

$$X <= i^*N - i^*(i^*+1)/2$$

which yields

$$i^* >= (N - 0.5) - \sqrt{N(N-1)Y + 0.25}$$

Hence let $\beta = \sqrt{N(N-1)Y + 0.25}$, then

$$N - 0.5 - \beta <= i^* <= N + 0.5 - \beta$$

or

$$i^* = \lfloor N + 0.5 - \beta \rfloor \quad , \qquad [9]$$

where $\lfloor a \rfloor$ denotes the greatest integer smaller than or equal to a.

Given this value for i^*, we draw a new random observation of Y ~ U(0,1) and rescale into X ~ U(i^*+1, N+1) by setting

$$X = Y(N-i^*) + i^* + 1$$

which in turn yields

$$j^* = \lfloor i^* + 1 + Y(N-i^*) \rfloor \qquad [10]$$

The corresponding arc (i^*,j^*) can now be deleted from the network provided that the conditions specified in Eq. [3] and Eq. [4] are satisfied. This procedure for deleting an arc is repeated until the network contains the desired number of arcs; i.e., until $\Sigma n_i = \Sigma m_j = A$.

3.2 The addition method

The Deletion Method will delete a total of N(N-1)/2 - A arcs. For certain values of N and A this may be a very time consuming process. In order to generate a network with N=4 and A=5 for example, the Deletion Method will have to delete one arc; however, if N=100 and A = 150, 4800 out of a total of 4950 arcs need to be deleted. Under certain conditions, considerable time savings may be obtained by using the **Addition Method**. As mentioned above, this procedure proceeds in the opposite direction; i.e., it starts from the adjacency matrix filled with zeros and adds the required number of ones.

As a consequence of the node labelling procedure adopted, there should always be an arc connecting nodes 1 and 2 and an arc connecting nodes N-1 and N. Consequently, the Addition Method will have to generate a total of A-2 arcs. This seems to suggest that a good heuristic strategy would be to use the Deletion Method if A > N(N-1)/4 and to use the Addition Method if otherwise.

Consider now the previous example with N=4 and A=5. Figure 4(a) represents the initial adjacency matrix with $a_{12} = 1$ and $a_{34} = 1$ according to the requirement that there should be at least one arc entering node 2 and one arc leaving node 4. The corresponding network is given in Figure 4(b).

****

Figure 4

****

The Addition Method will have to generate three additional arcs. It can be observed from Figure 4(b) that node 2 is not yet an emitting node and node 3 is not yet a receiving node. This means that of the three additional arcs to be generated, only one may be inserted arbitrarily since in the final network at least one arc must leave node 2 and at least one arc must enter node 3. In general the initial network will be characterized by $m = N-3$ non-receiving nodes and $n = N-3$ non-emitting nodes. This means that $f = A - 2 - m - n$ arcs may be generated and inserted in a purely random fashion.

Consequently, the Addition Method will start from the initial network and adjacency matrix (all $a_{ij} = 0$ except $a_{12} = 1$ and $a_{N-1,N} = 1$). It uses formulas [9] and [10] to generate an arc as long as the number of residual free arcs f is greater than zero, where

$$f = A - e - m - n \,, \qquad [11]$$

and initially, the number of generated arcs $e = 2$, the number of non-emitting nodes (nodes with zero outdegree) $n = N-3$, and the number of non-receiving nodes (nodes with zero indegree) $m = N-3$. Each time an arc is generated in this manner and checked for double selection, the adjacency matrix is updated and e is set to $e=e+1$. If the generated arc reduces the number of non-receiving nodes, we set $m = m-1$; if the number of non-emitting nodes is reduced, we set $n = n-1$.

If the residual number of free arcs $f <= 0$, we check if $m = 0$. If $m <> 0$, indicating that there is at least one non-receiving node, we locate the column j^* in the adjacency matrix that is completely filled with zeroes (if ties develop, take the highest column index). We generate a corresponding $i^* < j^*$, using equal probabilities; i.e.,

$$i^* = \lfloor 1 + (j^* - 1)\, Y \rfloor \,, \quad \text{where } Y \sim U(0,1) \qquad [12]$$

We add the corresponding arc (i^*,j^*), and update the adjacency matrix and the corresponding values of m, n and e.

If m = 0, we check if n, the number of non-emitting nodes equals zero. If n <> 0, we locate any zero row $i^* < N-1$ in the adjacency matrix and generate a corresponding node j^* using the formula

$$j^* = \lfloor i^* + 1 + (N - i^*) Y \rfloor \text{ , where } Y \sim U(0,1) \quad [13]$$

We add the corresponding arc (i^*,j^*) and update the adjacency matrix and the corresponding values of m, n and e.

We should realize, however, that the procedure described so far may generate a number of arcs $e > A$; i.e., generate more arcs than required. The following example illustrates this subtle point.

Let N = 6 and A = 7. Figure 5(a) denotes the initial adjacency matrix with a_{12} = 1 and a_{56} = 1 according to the requirement that there should be at least one arc entering node 2 and one arc leaving node 5. The initial network given in Figure 5(b), thus has n = N-3 = 3 non-emitting nodes and m = N-3 = 3 non-receiving nodes. This means that f = A-e-m-n = -1.

****
Figure 5
****

As long as m <> 0, the Addition Method must add arcs. Assume that it adds the three arcs (1,5), (1,4) and (1,3) in that order, according to the procedure described above. Now the number of non-receiving nodes m equals zero; i.e., m = 0, but n <> 0. The Addition Method will have to add arcs until there are no nodes left with zero outdegree; i.e., until n = 0. Assume that it adds the arcs (2,6), (3,6) and (4,6). This leads to the adjacency matrix and corresponding network given in Figure 6.

****

Figure 6

****

Up to now, however, a total of e = 8 arcs had to be generated for feasibility reasons, where the requirement was to generate a network with only A = 7 arcs. A normal way to proceed now is to use **the Deletion Method** to delete e-A arcs. The example in Figure 6 demonstrates that this will not be possible due to the violation of conditions [3] and [4] above. When situations like this arise; i.e., the Deletion Method is called from within the Addition Method to delete e-A arcs and the feasibility conditions do not allow arcs to be deleted, we randomly add an arc first and then use the Deletion Method to delete the required number of arcs. For the example given in Figure 6, the random addition of arc (2,4) for instance, would make it possible to delete arcs (1,4) and (2,6) without violating Eq. [3] and [4]. The result is a network with the required number of nodes and arcs.

3.3 The hybrid algorithm for generating a random activity network

The Deletion Method and Addition Method have been programmed in C for the IBM PS/2 Model 60 running under MS/DOS. The procedures have been validated using an extensive computational experiment. For each node value in the range N=4 to N=12, the number of arcs was varied from N to N(N-1)/2. For each node-arc combination so obtained, both procedures had to generate 100 networks. A comparison of the required CPU time indicated that the Deletion Method outperforms the Addition Method as long as

A > N(N-1)/4 + 2, while the Addition Method is to be preferred otherwise. Network structures for which the number of arcs to be generated equals the number of nodes were the most time consuming for both procedures. Table I gives the computational effort for these worst-case node-arc combinations.

****
Table I
****

Both the Deletion Method and Addition Method can be combined in a **hybrid procedure** for generating a random activity network. This hybrid algorithm can be described in pseudocode as follows.

GENERAL NETWORK

```
Read number of nodes N and number of activities A
IF  A >= N-1  AND  A <= N(N-1)/2
    IF A = N-1
        Generate a chain network
    ELSE
        IF  A > N(N-1)/4 + 2
            Initialize adjacency matrix
            DELETION METHOD to delete N(N-1)/2 - A arcs
        ELSE
            Initialize adjacency matrix
            ADDITION METHOD to add A-2 arcs
```

DELETION METHOD

```
WHILE not enough arcs deleted
    Generate (i*,j*) according to Eqs. [9]-[10]
    IF feasibility conditions Eqs. [3]-[4] satisfied
        Delete arc (i*,j*) and update
    ELSE (when called from within ADDITION METHOD)
        IF a sufficient large number of trials to delete
           an arc have failed
            WHILE no arc is added
                Generate random (i*,j*)
                IF arc does not exist
                    Add arc (i*,j*) and update
```

ADDITION METHOD

```
WHILE not enough arcs added
    WHILE  f > 0
        Generate (i*,j*) according to Eqs. [9]-[10]
        IF arc (i*,j*) not yet in network
            Add arc (i*,j*) and update
    WHILE f <= 0 and m+n > 0
        IF  m <> 0
            Locate a node j* with zero indegree
            Generate a node i* according to Eq. [12]
            Add arc (i*,j*) and update
        ELSE
            IF n <> 0
                Locate a node i* with zero outdegree
                Generate a node j* according to Eq. [13]
                Add arc (i*,j*)
IF  e > A
    DELETION METHOD to delete e-A arcs
```

The logic of the main routine **GENERAL NETWORK** and the two subroutines **ADDITION METHOD** and **DELETION METHOD** should be clear. If the requested number of arcs in the network is equal to N-1, the only possible structure for the network is a chain which is generated by setting the corresponding adjacency matrix elements equal to 1. If the requested number of arcs is greater than N(N-1)/4 + 2, the network will be generated using the DELETION METHOD in correspondance with the computational efficiency arguments made above; the ADDITION METHOD will be used otherwise. The selection branch following the ELSE in the DELETION METHOD only applies when the latter is called from within the ADDITION METHOD in order to delete the excessive e-A arcs, and this cannot be done without violating the feasibility constraints. An exhaustive check for the possibility to delete an arc can be very time consuming. That explains why we heuristically check if a sufficient large number of trials (greater than N(N-1)) to delete an arc have failed. If this is the case and no arc has been added yet, an arc (i^*,j^*) is randomly generated and the adjacency matrix is updated accordingly.

4. GENERATING A SET OF STRONGLY RANDOM ACTIVITY NETWORKS

In the previous section two procedures have been combined into an efficient hybrid procedure for generating a random feasible topological structure of an AN with a given number of nodes and arcs. However, as was mentioned earlier, many theoretical and practical situations require the use of a network generator for generating a set of strongly-random activity networks. This implies the generation of a set of (N,A) pairs, where for each pair several topological network structures and corresponding network data may be generated (see Elmaghraby and Herroelen (1980)).

4.1 The probability distribution of A given N

It was already argued above that for a given value of the number of nodes, N, the number of arcs, A, is limited by (N-1) <= A<= N(N-1)/2. Table II lists the number of feasible topological network structures for several (N,A) pairs. Since it would generally be too time consuming to generate all feasible network types for a given value of N, and even for a given (N,A) pair, a possible outcome would consist of determining the probability distribution of A given N, where for each value of N, a corresponding A-value can be obtained by drawing samples from the corresponding distribution.

****

Table II

****

It can be seen from Table II that for N <= 3, the p.d.f. has to assign equal probabilities to all feasible A-values. For N = 4 this equal probability assumption is no longer valid, but the p.d.f. of A given N is symmetric. For N > 4 , however, the p.d.f. is no longer symmetric but shows a skewness to the right which seems to increase with increasing N. Since obtaining an exact fit to this skew distribution announces itself as a cumbersome task (computing the number of topological structures itself is already onerous for large values of N), we opt for the following heuristic procedure.

Figure 7 plots the range of A for increasing N. The dotted curve represents the mean of the range of the number of arcs.

****

Figure 7

****

Since the values in Table II indicate that the observed mean of A lies below this theoretical mean, we have to adjust the latter. Therefore we set $l_A = N-1$ and $u_A = N(N-1)/2$ and compute the adjusted mean, μ_A, as follows:

$$\mu_A = (l_A + u_A)/2 - [(u_A - l_A)^2]/500 \qquad [14]$$

Setting

$$\sigma_A = (\mu_A - l_A)/2.5 \qquad [15]$$

and given a value of N, a corresponding A-value is obtained by drawing a sample from the normal distribution with adjusted mean, μ_A, and adjusted standard deviation, σ_A, as given by Eqs. [14] and [15] respectively.

Given the resulting (N,A) pair, the hybrid algorithm of the previous section may then be used to generate a random network structure.

4.2 A strongly-random activity network generator

The procedures described in the two previous sections have resulted in the construction of a strongly-random activity network generator. The software is available on diskette and can be run under MS-DOS on IBM PC/PS and compatibles. It interactively prompts the user to enter the number of nodes. The user may enter a specific value for **N**, opt for a random generation of the number of nodes by entering the relevant parameters for either the uniform, exponential, gamma, beta, normal, Poisson, binomial distribution, or define a node distribution of his own choice. The user is then prompted to specify the value for the number of activities **A**, either by direct entry of a specific number, or by invoking the procedure explained in Section 4.1 above. The hybrid procedure described in Section 3 can then be activated to generate a desired number of networks.

In addition the software allows for the random generation of the various network functions: activity durations (all equal or drawn from one of the precoded distributions); number of renewable resource types (limited to three); resource availabilities and requirements (equal or drawn from precoded distributions), and critical event cash flows.

5. SUMMARY AND CONCLUSIONS

Validation experiments in the field of activity networks require the generation of a set of strongly-random networks, where each network is characterized by a certain number of nodes and arcs, a random topological network structure and random values for the network functions. In this paper we present a hybrid combination of two efficient procedures, the Deletion and Addition Method, for generating a random activity-on-the-arc network with a given number of nodes and arcs. The hybrid procedure has been integrated into a computerized generator for a set of strongly-random networks, characterized by a representative range of the number of nodes and arcs, random network structures, and random values for the various network functions.

This network generator may proof to fill a need in many computational experiments aimed at measuring the network complexity (see Elmaghraby and Herroelen (1980)) or set up in order to validate optimal and suboptimal procedures for the many combinatorial problems which arise in the context of networks in general and activity-on-the-arc networks in particular.

REFERENCES

ALVAREZ-VALDES, R., "Computational Comparison of Classical and New Heuristic Algorithms for Resource-Constrained Project Scheduling", Paper presented at the First International Workshop on Project Management and Scheduling, Lisbon, 11-13 July 1988.

CHRISTOFIDES, N., R. ALVAREZ-VALDES and J.M. TAMARIT, "Project Scheduling with Resource Constraints: A Branch and Bound Approach", European J. of Ops. Res., 29,3 (1987), 262-273.

DODIN, B., "Approximating the Distribution Functions in Stochastic Networks", Comput. & Ops. Res., 12,3 (1985), 251-264.

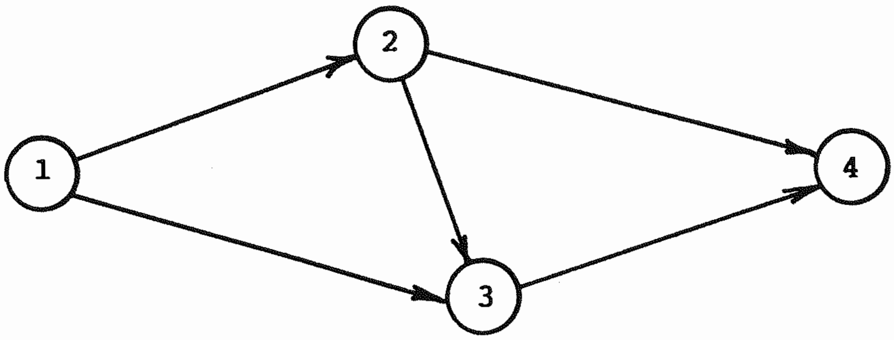
ELMAGHRABY, S.E. and W.S. HERROELEN, "On the Measurement of Complexity in Activity Networks", European J. of Ops. Res., 5 (1980), 223-234.

HERROELEN, W.S. & G. CAESTECKER, "The Generation of Random Activity Networks", Onderzoeksrapport N° 7906, Department of Applied Economic Sciences, K. U. Leuven, 1979.

KURTULUS, I. and E.W. DAVIS, "Multiproject Scheduling: Categorization of Heuristic Rules Performance", Management Sci., 28,2 (February 1982), 161-172.

PATTERSON, J., "A Comparison of Exact Approaches For Solving The Multiple Constrained Resource Project Scheduling Problem", Management Sci., 30, 7 (July 1984), 854-867.

TALBOT, F., "Resource-Constrained Project Scheduling With Time-Resource Tradeoffs: The Nonpreemptive Case", Management Sci., 28, 10 (October 1982), 1197-1210.



$$[a_{ij}] = \begin{bmatrix} 0 & 1 & 1 & 0 \\ & 0 & 1 & 1 \\ & & 0 & 1 \\ & & & 0 \end{bmatrix}$$

Figure 1. Typical activity network and corresponding adjacency matrix

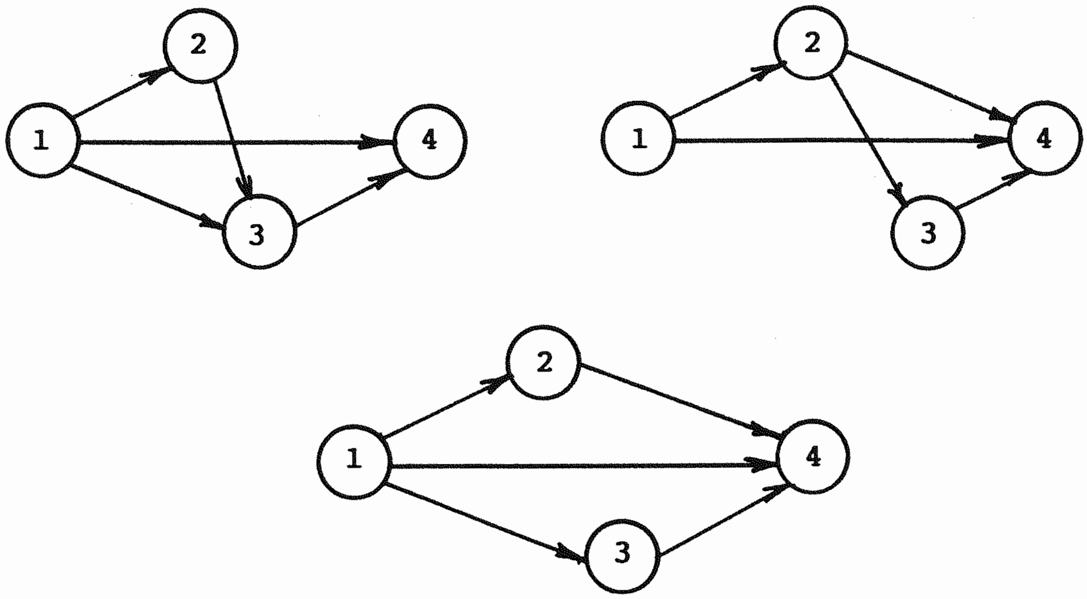


Figure 2. The remaining feasible topological network structures with $N = 4$ and $A = 5$.

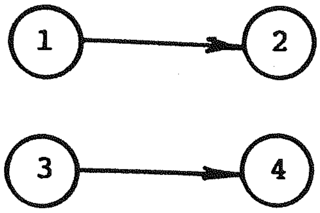
	1	2	3	4
1	0	1 (1)	1 (2)	1 (3)
2		0	1 (4)	1 (5)
3			0	1 (6)
4				0

Label	Range	Node
	0	
1		
	1/6	
2		i=1
	2/6	
3		
	3/6	
4		
	4/6	i=2
5		
	5/6	
6		i=3
	6/6	

Figure 3. Label and probability assignment.

$$[a_{ij}] = \begin{bmatrix} - & 1 & 0 & 0 \\ & - & 0 & 0 \\ & & - & 1 \\ & & & - \end{bmatrix}$$

(a) Initial adjacency matrix

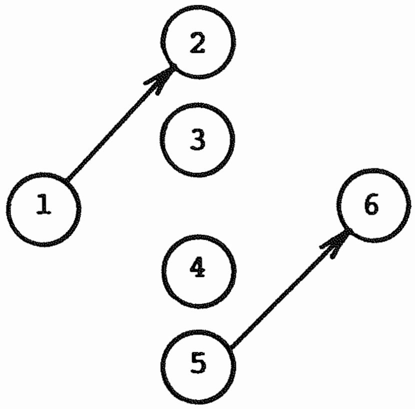


(b) Initial topological structure

Figure 4. Adjacency matrix and corresponding topological structure

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 \\ & & 0 & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & & 0 & 1 \\ & & & & & 0 \end{bmatrix}$$

(a) Adjacency matrix

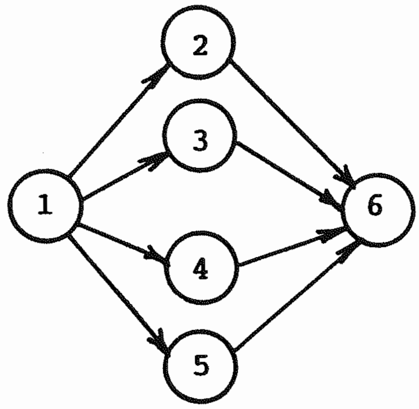


(b) Network

Figure 5. Adjacency and activity network for which e = 2, m = 3, n = 3 and f = -1

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ & 0 & 0 & 0 & 0 & 1 \\ & & 0 & 0 & 0 & 1 \\ & & & 0 & 0 & 1 \\ & & & & 0 & 1 \\ & & & & & 0 \end{bmatrix}$$

(a) Adjacency matrix



(b) Network

Figure 6. Adjacency matrix and network for which e = 8, m = 0, n = 0

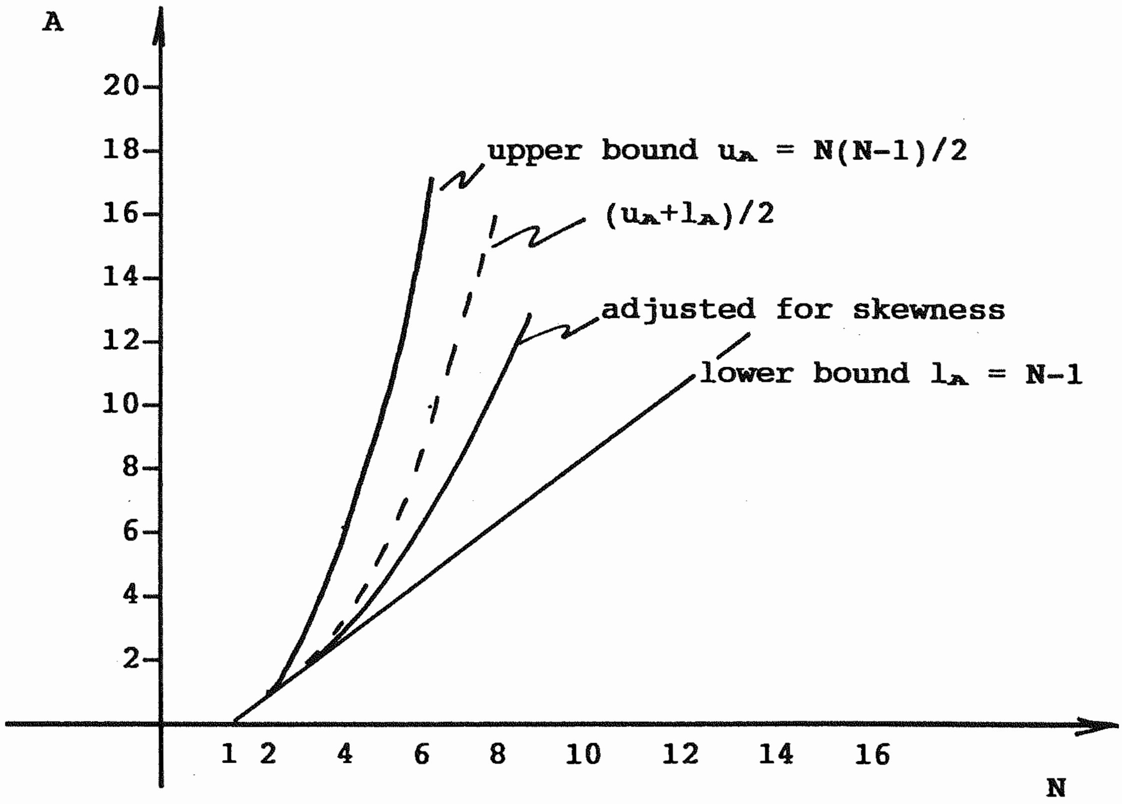


Figure 7. Range of A for increasing values of N

Table I. Computational results for the Deletion and Addition Method for worst-case node-arc combinations

Node-arc combination	CPU time in seconds (average for 100 networks) Deletion Method	Addition Method
N = 4 ; A = 4	0.02850	0.00720
N = 5 ; A = 5	0.09280	0.01650
N = 6 ; A = 6	0.32680	0.21250
N = 7 ; A = 7	1.38740	1.16830
N = 8 ; A = 8	4.57140	4.01780
N = 9 ; A = 9	8.93570	7.54120
N = 10 ; A = 10	31.88640	40.23370
N = 11 ; A = 11	76.80390	34.82010
N = 12 ; A = 12	103.26250	72.65410

Table II. The number of feasible network structures for several (N,A) pairs

N \ A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	1														
3		1	1												
4			1	4	4	1									
5				1	11	33	42	26	8	1					
6					1	26	171	507	840	865	584	262	76	13	1